

Routing in Wireless Networks with Interferences

Bogdan S. Chlebus* Vicent Cholvi† Paweł Garncarek‡

Tomasz Jurdziński§ Dariusz R. Kowalski¶

Abstract

We consider dynamic routing in multi-hop wireless networks with adversarial traffic. The model of wireless communication incorporates interferences caused by packets' arrivals into the same node that overlap in time. We consider two classes of adversaries: balanced and unbalanced. We demonstrate that, for each routing algorithm and an unbalanced adversary, the algorithm is unstable against this adversary in some networks. We develop a routing algorithm that has bounded packet latency against each balanced adversary.

Keywords: Wireless network, routing, adversarial queuing, interference, queue size, packet latency.

1 Introduction

Models of wireless data networks that abstract from incidental systems details and concentrate on the essential aspects of communication are most conducive to studying routing algorithms. One of such aspects are interferences. The model of *radio networks* [8] assumes that when multiple packets arrive simultaneously into a node then this results in interference experienced by the receiving node. Such networks are considered in this paper.

Adversarial methodologies of traffic generation make it possible to consider worst-case behavior of routing. We use such an approach to study routing in radio networks. Such networks pose unique challenges to design of routing algorithms because of the need to coordinate activities of the nodes whose transmissions may reach some node simultaneously.

Related work. The methodology of adversarial routing in wired networks was pioneered by Borodin et al. [5] and Andrews et al. [2]. Lotker et al. [13] showed

*Department of Computer Science and Engineering, University of Colorado Denver, Colorado, USA

†Departament de Llenguatges i Sistemes Informàtics, Universitat Jaume I, Castelló, Spain

‡Instytut Informatyki, Uniwersytet Wrocławski, Wrocław, Poland

§Instytut Informatyki, Uniwersytet Wrocławski, Wrocław, Poland

¶Department of Computer Science, University of Liverpool, United Kingdom

that, in wired networks, every greedy scheduling policy is stable if the injection rate is smaller than $1/(L + 1)$, where L is the length of the longest route used by any packet.

Stability in general wireless networks without explicit interferences was studied by Andrews and Zhang [3, 4] and Cholví and Kowalski [10]. Lim et al. [12] analyzed the stability of the max-weight protocol in wireless networks with interferences, but assuming the existence of a set of feasible edge rate vectors sufficient to keep the network stable.

Chlebus et al. [9] and Anantharamu et al. [1] studied adversarial broadcasting in the case of using single-hop radio networks. Chlebus et al. [7] considered interactions among components of routing in wireless networks, which included transmission policies, scheduling policies to select the packet to transmit from a set of packets parked at a node, and hearing control mechanisms to coordinate transmissions with scheduling.

Our results. We study dynamic routing in multi-hop radio networks with a specific methodology of adversarial traffic that reflects interferences. We demonstrate that there is no routing algorithm guaranteeing stability for an injection rate greater than $1/L$, where the adversary's parameter L is the largest number of links which a packet needs to traverse while routed to its destination. We give a routing algorithm that guarantees stability for injection rates smaller than $1/L$.

2 Routing against Interferences

We consider communication in multi-hop radio networks. A network is modeled as a (simple) undirected connected graph $G = (V, E)$ with some $n = |V|$ nodes. An edge in E represents two directed communication channels connecting the endpoints; an oriented edge from E is referred to as a *link*. An edge (u, v) , when interpreted as a link with tail u and head v , is denoted as $u \rightarrow v$. Messages are transmitted along the links according to the links' orientation. At most one link determined by an edge can be used at a time.

We say that some parameter of the communication environment is *known* when it can be used in an algorithm's code. Each node is assigned a unique name, which is an integer in $[1; n]$. Every node knows n and its own name.

An execution of a communication algorithm is synchronous, in that it is structured as a sequence of *rounds*. In each round, a node may either *transmit* a message or *listen* trying to hear incoming messages. Messages are delivered in the round of transmission. A message that is successfully received is said to be *heard* by the receiving node. A node v can hear a message from its neighbor u in a round t if v listens in round t and u is the only node among v 's neighbors that transmits in this round. Messages delivered simultaneously to a node but not heard by the node are said to *collide* or *interfere with one another* at the node. Messages facilitate routing, in particular they may carry packets traversing the network. Nodes may need to store multiple packets in their private memory, which is referred to as the node's *queue*. The number of

packets residing simultaneously in such a queue is the queue's *size*.

2.1 Routing

A routing algorithm handles packets that are injected at the nodes of graph G and need to reach their respective destination nodes by traveling through the network in a store-and-forward manner. A packet together with the round it was injected in and the (simple) oriented path it needs to traverse, make a *tour*. Each packet is encapsulated as a part of its tour at the time of injection and during the network's traversal. The number of links in a tour's path is this path's *length*, also referred to as the *tour's length*.

Consider a tour determined by a packet p injected in round t into node v_1 that needs to pass through the nodes on the path $\langle v_1, \dots, v_k \rangle$ to reach v_k . Each link $v_i \rightarrow v_{i+1}$ is said to be among the tour's *links*, for $1 \leq i < k$. The start node v_1 is this tour's *source* and the end node v_k is this tour's *destination*. Packet p is routed by traversing the links according to the tour's specification: there are $k - 1$ rounds t_1, \dots, t_{k-1} such that the node v_i transmits packet p in round t_i and the node v_{i+1} hears p in this very round t_i , for $i \in [1, k - 1]$, where $t \leq t_1 < t_2 < \dots < t_{k-1}$. The packet is *delivered* in round t_{k-1} , which is the round when the packet reaches the tour's destination. The *latency* of this specific routing of the tour is $t_{k-1} - t$, which is the number of rounds the tour spends in the network between its packet's injection at the source and its delivery to the destination.

2.2 Interferences

We represent interferences as abstract conflicts between parts of a network. The basic case is of a conflict between a node w with a link $u \rightarrow v$. Intuitively, it occurs when a transmission by w cannot be reconciled with having a different message delivered successfully from u to v in the same round. A node w *conflicts with a link* $u \rightarrow v$ if either $w = u$ or $w = v$ or the nodes v and w are neighbors. We extend this to say that a node w *conflicts with a tour* if w conflicts with some among the tour's links. This concept of conflict between a node and a tour conservatively reflects the worst-case possibility of the node's transmission interfering with the tour's packet when it is traversing the tour's path.

When transmissions through links of a tour may prevent packet deliveries along the links of another tour then these tours are said to be in conflict. Formally, tours f_0 and f_1 *conflict with one another* if either they pass through the same node or there is a node in one of these tours f_i , which is different from f_i 's destination, that conflicts with the other tour f_{1-i} . Conflicting pairs of tours can be interpreted as edges in a new graph. Formally, for a simple graph $G = (V, E)$ and a set of tours F in G , the *conflict graph* of F is a simple graph with tours in F taken as the vertices and two different tours from F making an edge when they conflict with each other.

Figure 1 gives an example of a conflict graph for a network and a set of tours in it, where arrows represent the traversed paths. The graph on the left

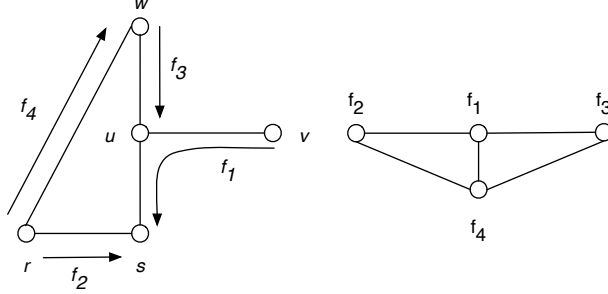


Figure 1: Four tours named f_1, f_2, f_3, f_4 . Tour f_1 conflicts with each of the other tours f_i , for $i \neq 1$, because it shares nodes with both f_2 and f_3 , and the start node r of f_4 conflicts with the link $u \rightarrow s$, which belongs to f_1 . Tour f_2 conflicts with f_4 because they share the node r , while f_2 does not conflict with f_3 because they do not share nodes and only their respective destination endpoints s and u conflict with the other tour. Tours f_3 and f_4 conflict with each other because they share the node w .

represents a network in which routing is performed, and the resulting conflict graph for the specified tours is depicted on the right.

2.3 The static link scheduling problem

We now consider static routing, when a set of routing tours is given as input. (This is an aside to the main topic of dynamic routing.) The problem is further restricted such that each tour is just one link. The goal is to route all these tours within the shortest possible interval of rounds. This is known as the *static link scheduling (SLS)* problem [11]. We show that SLS is related to vertex-colorings of conflict graphs. The minimum number of colors used in coloring vertices of a graph, such that each pair of adjacent vertices are assigned distinct colors, is the *chromatic number* of the graph.

Theorem 1. *The minimum number of rounds to route an instance of the link scheduling problem is equal to the chromatic number of the respective conflict graph.*

Proof. Consider a set F of one-link tours that makes an instance of the static link scheduling problem. Let μ be the chromatic number of the respective collision graph. Let T be the number of rounds of a shortest schedule S to route the packets in F . We want to show that $T = \mu$.

First we show that $\mu \leq T$. We may assume without loss of generality that each packet is transmitted exactly once in S . Namely, we may replace S by a minimal subset of transmissions in S such that each packet's transmission results in this packet's delivery to its destination. This implies that packets of

conflicting tours are transmitted in different rounds. Now assign the round of transmission of a packet as the color of the node representing this packet's tour. This is a proper coloring because an edge connects two conflicting tours, which are transmitted in different rounds. Therefore there exists a vertex coloring of the conflict graph of T colors, and so $\mu \leq T$.

Next we show that $T \leq \mu$. Consider a coloring of the conflict graph with μ colors. The colors could be identified with the integers in the interval $[1, \mu]$. A schedule of transmissions in μ rounds can be defined as follows: a node of color i transmits in round i . When a node transmits a packet then the packet is heard by the destination node because no packets with conflicting tours are transmitted in this round, as their tours are of different colors. It follows that all the packets get delivered in μ rounds, so that $T \leq \mu$. \square

2.4 Adversaries inject tours

We model dynamic injection of tours by way of an adversarial model, in the spirit of similar approaches used in [5, 2, 13, 9, 1, 10, 7]. An adversary represents the users that generate packets to be routed in a given radio network. The constraints imposed on packet generation by the adversary allow to consider worst-case performance of deterministic routing algorithms handling dynamic traffic.

An adversary is determined by three numbers (ρ, b, L) , which together are called its *type*. The number ρ is the *injection rate* and needs to satisfy $0 \leq \rho \leq 1$. The number $b > 0$ is called the *burstiness* and represents the maximum number of tours injected in the same round that may conflict with a node. The number L is the *stretch* and is a positive integer.

An adversary injects packets continuously by placing them in the nodes of a network. An injected packet is encapsulated with the path it needs to traverse into a tour.

Let τ be a time interval and v a node. We refer to the number of tours injected during τ that v conflicts with as the *load of node v in τ* . This means that a tour f contributes a unit to the load of each node v such that v conflicts with f .

The adversary of a given injection rate ρ , burstiness b and stretch L is subject to the following restrictions in how tours may be injected. First, for each time interval τ and each node v the load of v in τ is at most $\rho \cdot |\tau| + b$. Second, the length of path of an injected tour is at most L .

An adversary of type (ρ, b, L) is called *balanced* when the inequality $\rho \cdot L < 1$ holds, and it is *unbalanced* when the inequality $\rho \cdot L > 1$ holds.

3 Stability

A routing algorithm handles itineraries, which include the paths the packets are to traverse. A critical part of a routing algorithm is a *transmission policy* which determines which nodes transmit in a round, as well as the contents of

the transmitted messages. We consider distributed transmission policies, when each node decides in each round whether to transmit a message, and if so, then it determines the contents of the transmitted message. A message contains one tour stored in the node's queue, and possibly additional control bits.

The immediate goal of transmitting the packet of a tour is to forward it to the node designated in the tour as the next one on the path to be traversed by the tour's packet. A transmitted tour is heard by the intended recipient node when that recipient node is not transmitting in this round and the transmitted message does not interfere with other transmissions, following the radio network model's specification, as given in Section 2.

A routing policy is *stable* against an adversary when the number of tours in queues at the nodes is bounded in all executions when packet injections conform to the restrictions imposed by the adversary's type.

Packet latency of a routing algorithm against an adversary is the maximum latency attained by a tour in executions subject to the restrictions imposed by the adversary's type. When packet latency is bounded then queues are also bounded, as each queue size in a node is the lower bound on the delay of a packet already queued.

Theorem 2. *For each unbalanced adversary and each sufficiently large integer $n > 0$ there exists a network of n nodes such that every routing algorithm is unstable when the adversary injects tours into this network.*

Proof. Let (ρ, b, L) be the type of an unbalanced adversary, which means such that $\rho \cdot L > 1$. Take an arbitrary $n > L$ and let the network be the clique of n nodes.

An injected tour of a positive length contributes a unit to each node's load, because all nodes are neighbors. Therefore, the adversary can inject up to $\rho t + b$ tours into all the nodes in the network in a time interval of length t . When there are multiple disjoint time intervals of length t each, then the burstiness b can be accounted for at most once, but up to ρt new packets can be injected in each such an interval.

The adversary will inject packets that need to be forwarded exactly L times each. In a complete network, at most one message can be heard in a round, so at most one packet can be forwarded in a round. As each tour contains exactly L links, the total number of message to be heard, in order to deliver the packets injected in a time interval of length t , is at least

$$L\rho t = t + (L\rho - 1)t.$$

At most t messages can be heard in t rounds, so the adversary can generate, in disjoint intervals of t rounds, a surplus of $(L\rho - 1)t$ messages to be heard in the future. Take an integer t such that $(L\rho - 1)t \geq 1$, which exists because $\rho \cdot L > 1$.

At least one packet is needed to account for L messages. Thus the adversary can make the number of packets in the queues grow by at least one packet per L intervals in a sequence of consecutive disjoint time intervals of t rounds each. The resulting execution is unstable. \square

4 Efficient Routing

We specify a routing algorithm that we call OLD-GO-FIRST. It provides bounded packet latency when executed against balanced adversaries. In the algorithm's design, we rely on the Brook's theorem [6], which states that a graph of a maximum node degree Δ can be colored with $\Delta + 1$ colors.

The algorithm's execution is partitioned into disjoint intervals of rounds called *windows*. Tours injected in a window are *new* in this window and become *old* when the next window starts. Tours that are old in a window are transmitted in the window, while the new tours wait for the next window to be transmitted in it as old.

A window is partitioned into two phases, see Figure 2. The first phase consists of preprocessing in order to prepare the second phase, which is spent executing a transmission policy.

The phases are specified in greater detail next.

Phase 1: preprocessing to prepare routing in the next phase:

1. Collect in each node the specification of all the old tours. Let L' denote the length of the longest old tour.
2. Build the conflict graph for the old tours. Let Δ be the maximum degree of a node in the conflict graph.
3. Color the vertices of this conflict graph with $\Delta + 1$ colors.

Phase 2: tours are routed in the next time interval of $L'(\Delta + 1)$ rounds by the following transmission policy:

1. Partition this time interval into L' intervals, called *super-rounds*, each of $\Delta + 1$ rounds.
2. In each super-round: a node storing a tour of color i transmits this tour in the i th round of the super-round.

A node stores at most one tour of each color in the beginning of a super-round, because tours passing through a node conflict with each other, so they are colored differently. Collecting the information about all the old tours originating in each node can be accomplished by gossiping of what each node stores originally. We can use the algorithm given in [8], which runs in $S(n) = \mathcal{O}(n \log^4 n)$ rounds in a network of n nodes. This algorithm uses “short” messages, in that one rumor (the information to be gossiped that originates in one node) requires one message, and it takes one round to transmit a message.

For the purpose to prepare a transmission policy for a window, a rumor contains the information about all the old tours stored in a node at the end of the previous window. Once gossiping is completed, building the conflict graph and coloring its nodes [14] can be done in negligible time, concurrently by all the nodes.

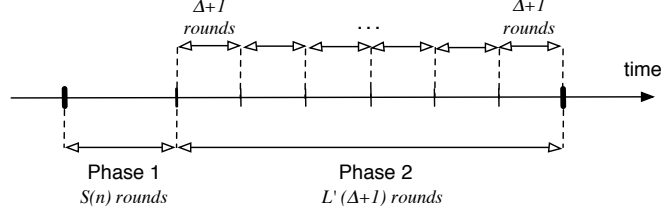


Figure 2: A window of an execution of transmission policy OLD-GO-FIRST consists of two phases. Phase one is preprocessing. Phase two consists of L' super-rounds, each of $\Delta + 1$ rounds, where Δ is the maximum degree of the conflict graph of the old tours in this window, which was built in phase one.

Theorem 3. *Routing algorithm OLD-GO-FIRST attains packet latency $\mathcal{O}\left(\frac{n \log^4 n + bL}{1 - \rho L}\right)$ when executed against a balanced adversary of type (ρ, b, L) on a network of n nodes.*

Proof. The number of tours injected in a window of length w that contribute to the load of a node is at most $\rho w + b$, each of stretch at most L . By the design of OLD-GO-FIRST, in a super-round, each tour, that is still on its way, becomes colored and so is transmitted at some round of this super-round. The message with such a tour is heard immediately by the receiving node, by the definition of coloring. Since phase two consists of L' super-rounds, each old tour is delivered to its destination within this window. Window size w needs to be sufficiently large to accommodate the two phases. By the design of the two phases, it is sufficient if w satisfies the following inequality:

$$S(n) + (\rho w + b) \cdot L \leq w. \quad (1)$$

This is because phase one takes $S(n)$ rounds, and $\Delta + 1 \leq \rho w + b$, as at most these many tours conflicting with a node can be injected during the previous window. Bound (1) is equivalent to $S(n) + bL \leq w(1 - \rho L)$, by algebra. Since the inequality $\rho L < 1$ holds, by the assumption that the adversary is balanced, we may take the following quantity

$$u = \left\lceil \frac{S(n) + bL}{1 - \rho L} \right\rceil$$

as an upper bound of every window w . Packet latency is at most $2u$, because a packet injected in the beginning of a window is delivered by the end of the next window. Using the estimate $S(n) = \mathcal{O}(n \log^4 n)$, we conclude that $\mathcal{O}\left(\frac{n \log^4 n + bL}{1 - \rho L}\right)$ is an upper bound on packet latency. \square

5 Conclusion

We proposed an adversarial framework to study stability of deterministic distributed routing algorithms in multi-hop wireless networks with interferences. It is representative enough to deny stability for sufficiently strong adversaries, namely, the unbalanced ones. We showed that there exists a deterministic distributed routing algorithm that provides bounded packet latency against balanced adversaries in all connected radio networks. This algorithm needs to know the size of the network, which is required in gossiping, but the adversary does not need to be known. Theorem 1 implies that achieving optimal packet latency for static instances of routing, in the case when packets need to make one hop only, is equivalent to finding the chromatic number of the respective conflict graph. One can show that conflict graphs have sufficiently expressive topologies to make the problem of their coloring NP-hard, as is the case for all simple graphs; we omit the details.

Acknowledgments: This work was supported by the Polish National Science Centre under grant DEC-2012/06/M/ST6/00459 and by the Spanish Ministry of Education Culture and Sport under grant PRX16/00086.

References

- [1] Lakshmi Anantharamu, Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Deterministic broadcast on multiple access channels. In *Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–5, 2010.
- [2] Matthew Andrews, Baruch Awerbuch, Antonio Fernández, Frank Thomson Leighton, Zhiyong Liu, and Jon M. Kleinberg. Universal-stability results and performance bounds for greedy contention-resolution protocols. *Journal of the ACM*, 48(1):39–69, 2001.
- [3] Matthew Andrews and Lisa Zhang. Scheduling over a time-varying user-dependent channel with applications to high-speed wireless data. *Journal of the ACM*, 52(5):809–834, 2005.
- [4] Matthew Andrews and Lisa Zhang. Routing and scheduling in multihop wireless networks with time-varying channels. *ACM Transactions on Algorithms*, 3(3):33, 2007.
- [5] Allan Borodin, Jon M. Kleinberg, Prabhakar Raghavan, Madhu Sudan, and David P. Williamson. Adversarial queuing theory. *Journal of the ACM*, 48(1):13–38, 2001.
- [6] Rowland Leonard Brooks. On colouring the nodes of a network. *Mathematical Proceedings of the Cambridge Philosophical Society*, 37(2):194–197, 1941.

- [7] Bogdan S. Chlebus, Vicent Cholvi, and Dariusz R. Kowalski. Universal routing in multi hop radio network. In *Proceedings of the 10th ACM International Workshop on Foundations of Mobile Computing, (FOMC)*, pages 19–28. ACM, 2014.
- [8] Bogdan S. Chlebus, Dariusz R. Kowalski, Andrzej Pelc, and Mariusz A. Rokicki. Efficient distributed communication in ad-hoc radio networks. In *Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP), Part II*, volume 6756 of *Lecture Notes in Computer Science*, pages 613–624. Springer, 2011.
- [9] Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Adversarial queuing on the multiple access channel. *ACM Transactions on Algorithms*, 8(1):5:1–5:31, 2012.
- [10] Vicent Cholvi and Dariusz R. Kowalski. Bounds on stability and latency in wireless communication. *IEEE Communication Letters*, 14:842–844, 2010.
- [11] Thomas Kesselheim. Dynamic packet scheduling in wireless networks. In *Proceedings of the 31st ACM Symposium on Principles of Distributed Computing (PODC)*, pages 281–290, 2012.
- [12] Sungsu Lim, Kyomin Jung, and Matthew Andrews. Stability of the max-weight protocol in adversarial wireless networks. *IEEE/ACM Transactions on Networking*, 22(6):1859–1872, 2014.
- [13] Zvi Lotker, Boaz Patt-Shamir, and Adi Rosén. New stability results for adversarial queuing. *SIAM Journal on Computing*, 33(2):286–303, 2004.
- [14] San Skulrattanakulchai. Δ -list vertex coloring in linear time. *Information Processing Letters*, 98(3):101–106, 2006.